HACC: Extreme Scaling and Performance Across Diverse Architectures

Salman Habib
Argonne National Laboratory
habib@anl.gov

Vitali Morozov
Argonne National Laboratory
morozov@anl.gov

Nicholas Frontiere
Argonne National Laboratory
nfrontiere@anl.gov

Hal Finkel
Argonne National Laboratory
hfinkel@anl.gov

Adrian Pope
Argonne National Laboratory
apope@anl.gov

Katrin Heitmann
Argonne National Laboratory
heitmann@anl.gov

ABSTRACT

Supercomputing is evolving towards hybrid and accelerator-based architectures with millions of cores. The HACC (Hardware/Hybrid Accelerated Cosmology Code) framework exploits this diverse landscape at the largest scales of problem size, obtaining high scalability and sustained performance. Developed to satisfy the science requirements of cosmological surveys, HACC melds particle and grid methods using a novel algorithmic structure that flexibly maps across architectures, including CPU/GPU, multi/many-core, and Blue Gene systems. We demonstrate the success of HACC on two very different machines, the CPU/GPU system Titan and the BG/Q systems Sequoia and Mira, attaining unprecedented levels of scalable performance. We demonstrate strong and weak scaling on Titan, obtaining up to 99.2% parallel efficiency, evolving 1.1 trillion particles. On Sequoia, we reach 13.94 PFlops (69.2% of peak) and 90% parallel efficiency on 1,572,864 cores, with 3.6 trillion particles, the largest cosmological benchmark yet performed. HACC design concepts are applicable to several other supercomputer applications.

Categories: Scalability, Peak performance

1. OVERVIEW: SIMULATING THE SKY

Cosmological surveys are our windows to the grandest of all dynamical systems, the Universe itself. Scanning the sky over large areas and to great depths, modern surveys have brought us a remarkably simple, yet mysterious, model of the Universe, whose central pillars, dark matter and dark energy, point to new, and even more fundamental discoveries. The pace of progress continues unabated – the next generation of sky surveys demand tools for scientific inference that far exceed current capabilities to extract information from observations.

The already important role of cosmological simulations is expected to undergo a sea change as the analysis of surveys moves over to an approach based entirely on forward models of the underlying physics, encompassing as well the complex details of survey measurements. Such an end-to-end paradigm, based on the ability to produce realistic 'universes' on demand, will stress the available supercomputing power to its limits.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.
SC'13 November 17 - 21 2013, , USA
Copyright 2013 ACM 978-1-4503-2378-9/13/11...$15.00.
http://dx.doi.org/10.1145/2503210.2504566.

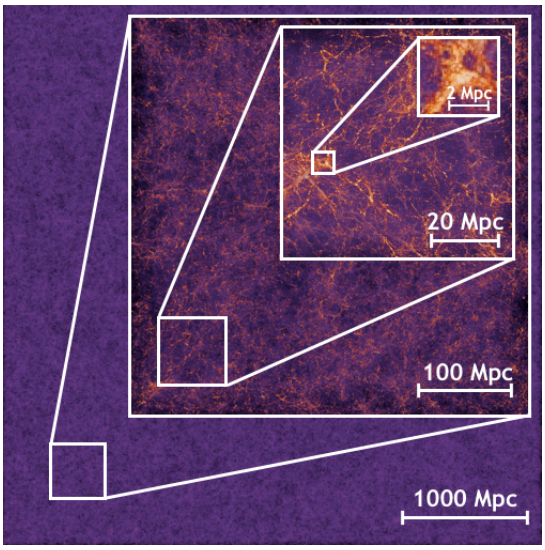

Figure 1: ***Zoom-in visualization of the density field in a 1.07 trillion particle,*** 4.25 ***Gpc box-size simulation with 6 kpc force resolution and particle mass,*** $m_p \sim 5 \cdot 10^9 M_\odot$***. This HACC simulation is currently running on 32 BG/Q racks. The image, taken during a late stage of the evolution, illustrates the global spatial dynamic range covered,*** $\sim 10^6$***, although the finer details are not resolved by the visualization.***

The desired improvements for simulations over the next decade are often stated in terms of orders of magnitude; high accuracy and robustness are central requirements to be met by this program. Because the simulations to be run are – and will continue to be – memory-limited on even the largest machines, stringent requirements must be simultaneously imposed on code performance and efficiency.

Cosmic structure formation at large scales is described by the gravitational Vlasov-Poisson equation [26], a 6-D PDE for the Liouville flow (1) of the phase space PDF where self-

consistency is imposed by the Poisson equation (2):

$$\partial_t f(\mathbf{x},\mathbf{p}) + \dot{\mathbf{x}}\cdot\partial_{\mathbf{x}} f(\mathbf{x},\mathbf{p}) - \nabla\phi\cdot\partial_{\mathbf{p}} f(\mathbf{x},\mathbf{p}) = 0, \quad (1)$$

$$\nabla^2\phi(\mathbf{x}) = 4\pi G a^2(t)\Omega_m\delta_m(\mathbf{x})\rho_c. \quad (2)$$

The expansion of the Universe is encoded in the time-dependence of the scale factor $a(t)$ governed by the cosmological model, the Hubble parameter, $H = \dot{a}/a$, G is Newton's constant, ρ_c is the critical density, Ω_m, the average mass density as a fraction of ρ_c, $\rho_m(\mathbf{x})$ is the local mass density, and $\delta_m(\mathbf{x})$ is the dimensionless density contrast,

$$\rho_c = 3H^2/8\pi G, \quad \delta_m(\mathbf{x}) = (\rho_m(\mathbf{x}) - \langle\rho_m\rangle)/\langle\rho_m\rangle, \quad (3)$$

$$\mathbf{p} = a^2(t)\dot{\mathbf{x}}, \quad \rho_m(\mathbf{x}) = a(t)^{-3} m \int d^3\mathbf{p} f(\mathbf{x},\mathbf{p}). \quad (4)$$

The Vlasov-Poisson equation is hopeless to solve as a PDE because of its high dimensionality and the development of nonlinear structure – including complex multistreaming – on ever finer scales, driven by the Jeans instability. Consequently, N-body methods, using tracer particles to sample $f(\mathbf{x},\mathbf{p})$ are used; the particles follow Newton's equations, with the forces on the particles given by the gradient of the scalar potential $\phi(\mathbf{x})$ [8].

Initial conditions given by a smooth Gaussian random field evolve into a 'cosmic web' comprised of sheets, filaments, and mass concentrations called halos [30, 35]. The first stars and galaxies form in halos and then evolve as the halo distribution also evolves by a combination of dynamics, mass accretion and loss, and by halo mergers. In addition to gravity, gasdynamic, thermal, radiative, and other processes must also be modeled. Large-volume simulations usually incorporate the latter effects via semi-analytic modeling.

Elementary arguments easily demonstrate the scale of the challenge to be overcome. Survey depths are of order a few Gpc (1 pc = 3.26 light-years); to follow bright galaxies, halos with a minimum mass of $\sim 10^{11}$ $M_\odot$ ($M_\odot$ = 1 solar mass) must be tracked. To properly resolve these halos, the tracer particle mass should be $\sim 10^8$ $M_\odot$ and the force resolution should be small compared to the halo size, i.e., ∼kpc. This immediately implies a dynamic range (ratio of smallest resolved scale to box size) of a part in 10^6 (∼Gpc/kpc) everywhere in the *entire* simulation volume (Fig. 1). In terms of the number of simulation particles required, counts range from hundreds of billions to many trillions. Time-stepping criteria follow from a joint consideration of the force and mass resolution [29]. Finally, stringent requirements on accuracy are imposed by the very small statistical errors in the observations – certain quantities such as lensing shear power spectra must be computed at accuracies of a *fraction* of a percent [17].

For a cosmological simulation to be considered "high-resolution", *all* of the above demands must be met. In addition, throughput is a significant concern. Scientific inference from cosmological observations defines a statistical inverse problem where many runs of the forward problem are needed to estimate cosmological parameters and associated errors. For such analyses, hundreds of large-scale, state of the art simulations will be required [20].

The HACC framework meets these exacting conditions in the realm of performance and scalability across a variety of node-level architectures. HACC has attained unprecedented sustained levels of performance, reaching up to 69% of peak on certain BG/Q partition sizes as well as scaling to the full size of Titan, with a similar time to solution.

2. CURRENT STATE OF THE ART

N-body simulations in cosmology have a long history, starting with Peebles' simulations of 300 particles in 1969 [25], to today's largest simulations evolving more than a trillion particles. Initial N^2 approaches gave way quickly to more efficient methods. Particle-mesh (PM) methods proved insufficient to obtain the required force resolution and were replaced by P^3M (Particle-Particle PM) algorithms (e.g., Ref. [7]), and tree codes [27]. Because of the high degree of clustering in cosmological simulations, P^3M codes have been mostly displaced by tree codes (nevertheless, as demonstrated by HACC, P^3M can be resurrected for CPU/GPU systems). For their work on this front, Gordon Bell Prizes were awarded to Warren and Salmon [34]. To localize tree walks and make handling periodic boundary conditions easier, hybrid TreePM methods were introduced and form the mainstay of gravity-only cosmology simulations. An early implementation is described in Ref. [37]; the most popular code used today, GADGET-2 [32], is also a TreePM code.

At the same time as the tree-based methods were being developed, high-resolution grid-based PM approaches using AMR (Adaptive Mesh Refinement) also made their appearance, e.g., Refs. [5, 10, 33]. AMR methods are particularly important in applications where gasdynamics and other astrophysical effects have to be included. In the case of large-volume survey simulations, the efficiency of AMR methods is reduced because clustering occurs over the entire simulation volume, leading to high AMR computational and memory costs (in turn degrading the available force and mass resolution). Consequently, the most successful AMR applications have involved the study of a smaller number of objects such as clusters of galaxies at high resolution and with different physics modules employed. For a comparison and overview of ten different codes spanning all algorithms discussed above see Ref. [19].

The number of fully equipped cosmology codes implemented on GPU systems is quite limited [23]. Two examples are ChanGA (CHARM++ N-body Gravity Solver) [23], a tree implementation and GAMER (GPU-accelerated Adaptive-MEsh-Refinement code) [31], an AMR code including both gravity and hydrodynamics solvers. ChanGA ran on up to 256 GPUs evolving up to 80 million particles in a 90 Mpc box [23]. GAMER was run on up to 32 GPUs, using a 100 h^{-1}Mpc box and a 256^3 root grid. Five levels of refinement imply a resolution of 12 h^{-1}kpc in the highest density regions [31]. These simulations are of modest size, in part due to the limited availability of very large GPU accelerated machines.

Next we discuss three examples of recent, very large cosmological simulations. It is difficult to compare performance metrics across codes if they are not run with the same specifications (e.g., accuracy) and on the same machine. In particular, the mass and force resolution play a crucial role in determining the time to solution. In very large volumes, with low force resolution, fewer dense structures form, and the computational effort is significantly reduced. In addition, accuracy requirements dictate the number of time steps as well as other criteria in designing the time-stepper. All of these different choices can change the computational effort and time to solution by large factors. These caveats must be kept in mind when considering the examples below.

Currently the largest fully analyzed high-resolution cosmological simulation is the Millennium XXL run [3], car-

ried out with the TreePM code GADGET-3. This simulation ran $\sim$ 300 billion particles in a 4.1 Gpc box, with a particle mass of $m_p = 8.5 \cdot 10^9 \mathrm{M}_\odot$, and a force resolution of 13.7 kpc. The simulation was carried out on the JuRoPa machine at the Jülich Supercomputer Center in 2010. A hybrid MPI/OpenMP programming model was used on 1536 compute nodes and 12888 cores. Overall, 87 trillion force calculations were performed at a cost of 2.86 Million CPU hours. GADGET-3 employs an adaptive time stepping scheme, therefore force calculations are not carried out for all particles at each time step; only a modest number of global PM time-steps were used (256).

Two simulations with more particles but significantly worse resolution are described in Refs. [24] and [2]. The Horizon Run 3 [24] evolved 374 billion particles in a 15 Gpc box using the TreePM code GOTPM [9] with a particle mass of $m_p = 3.4 \cdot 10^{11} \mathrm{M}_\odot$, 40 times worse than Millennium XXL. The simulation was run with 600 time steps with a somewhat coarse force resolution of 222 kpc. Detailed performance numbers were not provided, but it is difficult to compare this with the Millennium XXL run (and HACC timings), because of the much poorer mass and force resolution. The DEUS (Dark Energy Universe simulation) run by Alimi et al., was carried out in 2012 with the AMR code RAMSES [33], and ran 550 billion particles in a $\sim$ 30 Gpc box with an even lower level of mass resolution, $m_p \sim 4 \cdot 10^{12}$ $\mathrm{M}_\odot$. The base grid was 8192^3 and 6 levels of refinement led to a 55 kpc force resolution. The simulation was on run on 4752 nodes on the petascale system CURIE for 10 million CPU hours.

Finally, we briefly mention the TreePM simulation by Ishiyama et al., awarded the 2012 Gordon Bell Prize. They achieved $\sim$ 5 PFlops for a 1 trillion particle simulation on 82944 nodes of the K machine, a factor of 3 less than HACC's performance on Sequoia. The box size was 600 parsecs (note that cosmological boxes usually span of the order of Gigaparsecs) with initial and final redshifts, $z_{\mathrm{in}} = 400$ to $z_{\mathrm{fin}} = 31$. This computation covers a completely different regime than that characteristic of a true cosmological simulation and is therefore very difficult to compare to results from other codes. The limited size of the PM grid (4096^3) used is due to the fact that their FFT is slab-decomposed, and consequently, is not scalable.

No cosmological N-body code besides HACC has, to our knowledge, scaled to millions of cores or run at extreme levels of performance and efficiency at similar scales on a diverse set of architectures, at the same time allowing for strict accuracy controls, verified by using different force algorithms. This is in part due to FFT restrictions for other codes or due to complex data structures (such as those in AMR codes), leading to high memory demand per core and difficult to manage communication patterns, or restrictions arising from other aspects of algorithm or code design.

3. HACC: FACING THE MULTI-ARCHITECTURE CHALLENGE

3.1 Architectures and Algorithms: Two-Level Approach

A modern code framework has to confront issues raised by diverse architectures and programming models, as well as respond to disruptive changes in the future. Therefore it must have the ability to gracefully incorporate multiple algorithms that interoperate with each other, to optimize them for the architecture at hand, and place minimal reliance on external resources that can potentially limit these abilities. This is a concise statement of the design philosophy adhered to by HACC.

The realization of this strategy follows a two-level paradigm. As discussed in Section 2, the cosmological N-body problem is typically treated by a mix of grid and particle-based techniques. Grid-based techniques are better suited to larger ('smooth') length scales, with particle methods having the opposite property. This suggests that the higher level of code organization should be grid-based, interacting with particle information at a lower level of the computational hierarchy. Following this central idea, HACC uses a hybrid parallel algorithmic structure, splitting the gravitational force calculation into a specially designed grid-based long/medium range spectral particle-mesh (PM) component that is essentially architecture-independent, and an architecture-tunable particle-based short/close-range solver (Fig. 2). The grid is responsible for 4 orders of magnitude of dynamic range, while the particle methods handle the critical 2 orders of magnitude at the shortest scales where particle clustering is maximal and the bulk of the time-stepping computation takes place.

The flexibility to respond to different nodal architectures is built into the short/close-range solvers; these can use direct particle-particle interactions, i.e., a $\mathrm{P^3M}$ algorithm [21], as on Roadrunner and Titan, or use both tree and particle-particle methods as on the IBM BG/Q ('PPTreePM'). Access to multiple algorithms within the HACC framework also enables careful error testing and validation, for example, the $\mathrm{P^3M}$ and the PPTreePM versions agree to within 0.1% for the nonlinear power spectrum test in the code comparison suite of Ref. [18].

HACC's multi-algorithmic structure attacks several common weaknesses of conventional particle codes including limited vectorization, indirection, complex data structures, lack of threading, and short interaction lists. It combines MPI with a variety of nodal programming models (e.g., OpenCL, OpenMP) to readily adapt to different platforms. HACC has been ported to conventional and Cell/GPU-accelerated clusters, Blue Gene systems, and Intel Xeon Phi systems. HACC is the first, and currently the only large-scale cosmology code suite world-wide, that can run at scale (and beyond) on *all* available supercomputer architectures. To showcase this flexibility, we present scaling results for two very different cases in Section 4: the IBM BG/Q systems Mira at ALCF and Sequoia at LLNL, and Titan at OLCF.

3.2 Key Ideas and Innovations I: HACC Top Level

The top level of HACC's architecture consists of the domain decomposition, the medium/long-range force solver, and the interface to the short-range solver. All three aspects of this design involve new ideas to enhance flexibility and performance.

The spatial domain decomposition is in regular (non-cubic) 3-D blocks, but unlike guard zones in a typical PM method, full particle replication – which we term 'particle overloading' – is employed across domain boundaries. Overloading provides two crucial benefits. The first is that medium/long-range force calculations require no particle communication, and high-accuracy local force calculations require only sparse

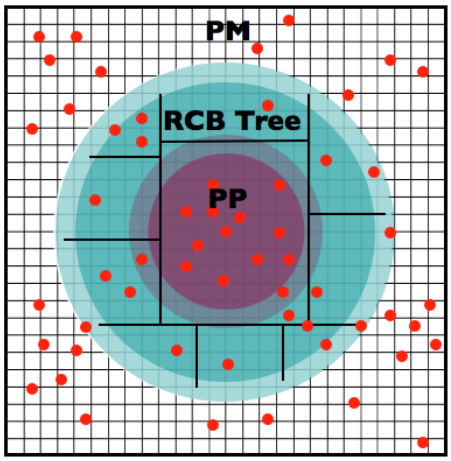

Figure 2: ***Informal representation of the HACC force evaluation hierarchy – 1) long/medium-range contributions from a high-order grid-based, spectrally filtered particle-mesh (PM) solver, 2) medium/short-range contributions using a (rank-local) recursive coordinate bisection (RCB) tree algorithm (green region), 3) close-range contributions using direct particle-particle (PP) interactions (magenta). Parameters governing the cross-overs are discussed in the text.***

refreshes of the overloading zone (for details, see Ref. [14, 28]). As is perhaps needless to emphasize, these features will acquire added significance for future computational node architectures.

The second major advantage of overloading is that it frees the local force solver from handling communication tasks, which are taken care of completely at the top level. Thus new 'on-node' local methods can be plugged in easily with guaranteed scalability, requiring only local optimizations. All short-range methods in HACC are local to the MPI-rank and the locality can be fine-grained further. This can be used to lower the number of levels in tree algorithms and to parallelize across fine-grained particle interaction sub-volumes. The benefits bestowed by overloading come at only modest cost: the memory overhead for a large run is only $\sim 10\%$.

The long/medium range algorithm is based on a fast, spectrally filtered PM method incorporating several sophisticated features. The density field is generated from the particles using a Cloud-In-Cell (CIC) scheme [21] and is then smoothed with an isotropizing spectral filter. The spectral filter reduces the anisotropy "noise" of the CIC scheme by over an order of magnitude without recourse to inflexible higher-order spatial particle deposition methods that are more commonly used. The noise reduction allows matching of the short and longer-range forces at small scales and confers the ability to use higher-order methods, both of which have important ramifications for accuracy and performance.

The Poisson solver uses a sixth-order influence function (spectral representation of the inverse Laplacian) [12]. The gradient of the scalar potential is obtained using low-noise fourth-order Super-Lanczos spectral differencing [15]. The "Poisson-solve" in HACC is the composition of all the kernels discussed above within one single discrete sum, each component of the potential gradient requiring an independent FFT. HACC uses its own scalable, high performance 3-D FFT routine implemented using a 2-D pencil decomposition (Section 4.)

3.3 Key Ideas and Innovations II: Short-Range Solvers

The form of the short-range force is given by subtracting the filtered grid force from the exact Newtonian force. The filtered force was determined to high accuracy using randomly sampled particle pairs and then fitted to an expression with the correct large and small distance asymptotics. Thanks to the effective use of filtering (Section 3.2), this functional form is needed only over a small, compact region, and can be represented using a fifth-order polynomial expansion, resulting in the crucial ability to vectorize computations in the main force kernel.

For heterogeneous systems such as Titan, the long/medium-range solver operates at the CPU layer. Depending on the CPU/accelerator memory balance, two different modes may be used, 1) grids held on the CPU and particles on the accelerator, or, more commonly, 2) a streaming paradigm, with grid and particle information resident in CPU memory, and short-range computations streamed through the accelerator. In both cases, the local force solve is a direct particle-particle interaction, resulting in a hardware-accelerated P^3M code. The accelerated short-range algorithm on Titan outperforms the corresponding CPU-only TreePM version by more than an order of magnitude.

The streaming of particles is carried out by partitioning the 3-D particle domain into 2-D data slabs. The slab width is between 3-4 grid cells, as the force resolution of the top level PM solver suffices for larger scales. Therefore, the short range force calculation on one slab only requires data from adjacent slabs. We dynamically store four slabs in the GPU memory at any given time, performing the P^3M algorithm on the middle slab. While the GPU performs its calculation on the chosen slab, the CPU host code simultaneously reads in the data results of the previous slab calculation, while writing the upcoming slab into GPU memory for later computation. The latency of reading and writing memory to the GPU is absorbed primarily by the GPU computation time as such memory movement can be performed simultaneously. This memory partitioning, not only eliminates the low memory problem of the GPU, but also drastically reduces the cost of memory movement between the CPU and the GPU, a pernicious performance chokepoint in GPU codes. In fact, each iteration (i.e. computation of the middle slab) takes longer then the simultaneous memory push, eliminating extra time spent on memory movement.

For a many-core system (e.g., BG/Q or Intel Xeon Phi), the GPU strategy is obviously not applicable, and it is more efficient to change the short-range solver to a tree-based algorithm. HACC uses a recursive coordinate bisection (RCB) tree in conjunction with a highly-tuned short-range polynomial force kernel. The implementation of the RCB tree, although not the force evaluation scheme, generally follows the discussion in Ref. [11]. Two core principles underlie the high performance of the RCB tree's design.

Spatial Locality. The RCB tree is built by recursively dividing particles into two groups, placing the dividing line at the center of mass coordinate perpendicular to the longest side of the box. Particles are then partitioned such that particles in each group occupy disjoint memory buffers. Local forces are computed one leaf node at a time. The particle data exhibits a high degree of spatial locality after the tree build; because the computation of the short-range force on the particles in any given leaf node, by construction, deals

with particles only in nearby leaf nodes, the cache miss rate is extremely low.

Walk Minimization. In a traditional tree code, an interaction list is built and evaluated for each particle. The tree walk necessary to build the list is relatively slow because it involves complex conditional statements and "pointer chasing" operations. A direct N^2 force calculation scales poorly as N grows, but for a small number of particles, a thoughtfully-constructed kernel can still finish the computation in a small number of cycles. The RCB tree exploits our highly-tuned force kernels to reduce the overall evaluation time by shifting workload away from the slow tree-walking and into the force kernel. On many systems, tens or hundreds of particles can be in each leaf node before the critical crossover point in computational efficiency is reached.

3.4 Key Ideas and Innovations III: Other Features

The time-stepping in HACC is based on a 2nd-order split-operator symplectic scheme that sub-cycles the short/close-range evolution within long/medium-range 'kick' maps where particle positions do not change but the velocities are updated. The number of sub-cycles can vary, depending on the force and mass resolution of the simulation, from $n_c = 5-10$. Local density estimates automatically provided by the RCB tree are used to enable adaptive time-stepping at the level of an individual leaf. HACC uses mixed precision computation – double precision is used for the spectral component of the code, whereas single precision is adequate for the short/close-range particle force evaluations and particle time-stepping.

As emphasized above, HACC's performance and flexibility are not dependent on vendor-supplied or other high-performance libraries or linear algebra packages; the 3-D parallel FFT implementation in HACC couples high performance with a small memory footprint as compared to available libraries. Unlike some other N-body codes that have done well in the Gordon Bell arena, no special hardware use is associated with HACC, and assembly-level programming is not required.

To summarize, the HACC framework integrates multiple algorithms and optimizes them across architectures; it has several performance-enhancing features, e.g., overloading, spectral filtering and differentiation, mixed precision, compact local trees, and locally adaptive time-stepping. Finally, weak scaling is a function only of the spectral solver; HACC's 2-D domain decomposed FFT guarantees excellent performance and scaling properties (Section 4).

4. PERFORMANCE

4.1 Performance Measurement Tools

In this paper, we focus on providing results for Titan and the IBM BG/Q architectures. Explicit performance results are given only for the BG/Q since no hardware counters are available for NVIDIA Kepler K20X GPUs. Therefore for Titan, our results primarily relate to wall clock measurements. For the BG/Q systems, performance was measured using IBM's internal use performance tool provided by Bob Walkup. This tool has two components – an MPI profile component, based on the PMPI interface and a Hardware Performance Monitor (HPM) component, based on a low-level API to access the Blue Gene Performance Monitor (BGPM) hardware.

The HPM tool integrates with HACC by serving as an MPI wrapper library, transparently collecting performance data using the BG/Q's hardware performance counters, aggregating it over all ranks, and writing summary files while HACC is exiting. HACC also makes use of a simple API provided by HPM to define a separate virtual set of counters for certain subregions of HACC's execution. This is used to collect performance data only in the main evolution loop, filtering out unwanted influences from job startup and I/O. HACC also uses this API to force the creation of specific summary files at the conclusion of every time step.

While HPM is capable of using a wide array of the available BG/Q hardware performance counters, HACC almost always uses the default configuration. This allows us to see data on instruction throughput, with distinct counters for floating-point operations, and hit/miss statistics from all levels of the memory hierarchy. MPI-level timing information is also provided, allowing us to identify load imbalance and expensive synchronization points.

4.2 Target Architectures and Environments

The defining characteristic of HACC, as already discussed, is its ability to run on diverse architectures (multi/many-core as well as heterogenous) *without sacrificing performance or scalability.* We will showcase this on two very different architectures: the GPU-accelerated system Titan and the BG/Q systems Sequoia and Mira. These machines currently occupy rank two, three, and five in the Top 500 list (http://www.top500.org/). These architectures represent two very different approaches to parallel supercomputing, a smaller number of "hot" nodes with a larger flops/bandwidth imbalance (Titan) versus a larger number of lower compute intensity nodes, with a more balanced network configuration (Mira and Sequoia). It is important to note that while code ports to Titan have involved a fair degree of effort (measured in man-years), the transition of HACC to Titan took less than a two-person month.

Titan, a hybrid Cray XK7 system, is the third generation of major capability computing systems at the Oak Ridge Leadership Computing Facility (OLCF). It is an upgrade of the original Jaguar system first installed at the OLCF in 2008. The initial upgrade from Cray XT5 to Cray XK6 compute nodes was accepted in February 2012 and consisted of 18,688 compute nodes for a total of 299,008 AMD Opteron 6274 "Interlagos" processor cores and 960 NVIDIA X2090 "Fermi" Graphical Processing Units (GPU). The peak performance of the Opteron cores is 2.63 PFlops and the peak performance of the GPUs is 638 TFlops in double precision. In late 2012, the 960 NVIDIA X2090 processors were removed and replaced with 18,688 of NVIDIA's next generation Tesla K20X "Kepler" processors, with a total system peak performance in excess of 27 PFlops in double precision.

The BG/Q is the third generation of the IBM Blue Gene line of supercomputers [22]. The BG/Q Compute chip (BQC) combines CPUs, caches, network, and a messaging unit on a single chip [16]; each BG/Q node contains the BQC and 16 GB of DDR3 memory. Each BQC uses 17 augmented 64-bit PowerPC A2 cores with specific enhancements: 1) 4 hardware threads and a SIMD quad floating point unit (Quad Processor eXtension, QPX), 2) a sophisticated L1 prefetching unit with both stream and list prefetching, 3) a wake-up unit to reduce certain thread-to-thread interac-

tions, and 4) transactional memory and speculative execution. Of the 17 BQC cores, 16 are for user applications and one for system services. Each core has access to a private 16 KB L1 data cache and a shared 32 MB multi-versioned L2 cache connected by a crossbar. The A2 core runs at 1.6 GHz and the QPX allows for 4 FMAs per cycle, translating to a peak performance of 204.8 GFlops for the BQC chip. The BG/Q network has a 5-D torus topology; each node has 10 communication links with a peak total bandwidth of 40 GB/s [6]. Our results have been obtained on Sequoia, a 96 rack system (1,572,864 cores) with ∼20 PFlops peak performance and on Mira, a 48 rack system (786,432 cores) with ∼10 PFlops peak performance.

4.3 Performance Results

The results are presented in three parts: performance and scaling of (i) the FFT and hence of the medium/long range solver, and of the full code on (ii) Titan, and (iii) on BG/Q systems. Weak and strong scaling results are shown for all cases. For the full code runs, the particle mass is $\sim 5 \cdot 10^{10} M_{\odot}$ and the force resolution, 6 kpc. All simulations are for a ΛCDM (Cold Dark Matter) model with $\Omega_m = 0.265$. Simulations of cosmological surveys focus on large problem sizes, therefore the weak scaling properties are of primary interest.

The full code exhibits essentially perfect weak scaling out to 16,384 nodes of Titan (∼90% of the system) at 92.2% parallel efficiency. It strong scales up to almost half of Titan on a problem with (only) 1024^3 particles. The biggest run with 1 trillion particles ($10{,}240^3$), far exceeds the largest cosmology run ever performed on GPU-accelerated systems.

On the BG/Q systems, HACC weak scales to the full machine size, achieving a performance of 13.94 PFlops on 96 racks, at around 67-69% of peak in most cases (up to 69.37%) at an efficiency of 90%. We demonstrate strong scaling up to one rack on a 1024^3 particle problem. Finally, the biggest test run evolved more than 3.6 trillion particles ($15{,}360^3$), exceeding by more than an order of magnitude the largest high-resolution cosmology run performed to date. Extensive details about the performance results on the BG/Q systems and how the high peak performance was achieved are given in Ref. [13]. Here we give a summary of those results.

4.3.1 *Scaling of the FFT and the Long/Medium-Range Solver*

The weak scaling of HACC is controlled by the FFT that underlies the spectral Poisson solver (Section 3). To achieve extreme scalability, HACC has its own fast, portable, and memory-efficient pencil-decomposed, non-power-of-two FFT (data partitioned across a 2-D subgrid), allowing $N_{rank} < N^2_{FFT}$, sufficient for use in any supercomputer in the foreseeable future. The FFT is composed of interleaved transposition and sequential 1-D FFT steps, where each transposition only involves a subset of all tasks; the transposition and 1-D FFT steps are overlapped and pipelined, with a reduction in communication hotspots in the interconnect. Details of the implementation are rather complex, requiring careful scheduling of communication phases in order to avoid deadlock.

Timing information for the FFT on the BG/Q and Titan is given in Table 1. The first (top) part of the table shows results from a strong scaling test for a fixed FFT size of 1024^3. As ranks increase from 256 to 8192 (8 ranks per

Table 1: FFT scaling up to 10240^3 grid points on the BG/Q (column 3/4) and Titan (column 5/6); time measured in seconds. First block: strong scaling results for both machines, second block: weak scaling with $\sim 200^3$ grid points/core on Mira, third block: weak scaling with $\sim 320^3$ grid points/per rank on Titan, last row: 400^3 grid points/rank.

Size S	Ranks R	BG/Q Time T	BG/Q R*T/S	Titan Time T	Titan R*T/S
1024^3	256	2.731	$6.51 \cdot 10^{-7}$	2.473	$5.89 \cdot 10^{-7}$
1024^3	512	1.392	$6.63 \cdot 10^{-7}$	1.291	$6.15 \cdot 10^{-7}$
1024^3	1024	0.713	$6.80 \cdot 10^{-7}$	0.607	$5.79 \cdot 10^{-7}$
1024^3	2048	0.354	$6.75 \cdot 10^{-7}$	0.306	$5.83 \cdot 10^{-7}$
1024^3	4096	0.179	$6.83 \cdot 10^{-7}$	0.173	$6.59 \cdot 10^{-7}$
1024^3	8192	0.098	$7.48 \cdot 10^{-7}$	0.093	$7.10 \cdot 10^{-7}$
5120^3	16384	10.36	$12.65 \cdot 10^{-7}$		
6400^3	32768	12.40	$15.55 \cdot 10^{-7}$		
8192^3	65536	14.72	$17.55 \cdot 10^{-7}$		
10240^3	131072	14.24	$17.40 \cdot 10^{-7}$		
5120^3	4096			22.67	$6.92 \cdot 10^{-7}$
6656^3	8192			26.40	$7.33 \cdot 10^{-7}$
8192^3	16384			27.06	$8.06 \cdot 10^{-7}$
10240^3	16384			52.19	$7.96 \cdot 10^{-7}$

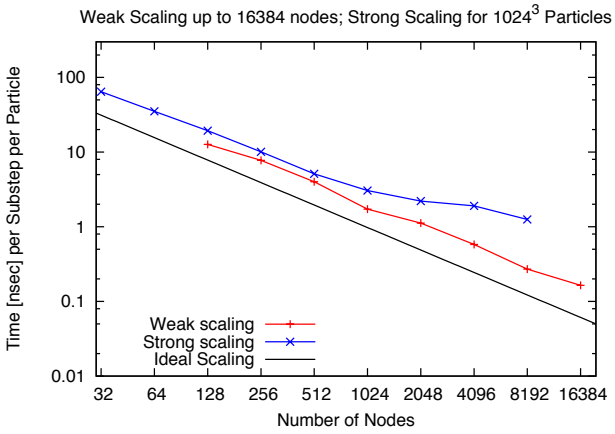

Figure 3: ***Weak and strong scaling on Titan. Weak scaling (red) is reported for ∼32 million particles per node. The time per substep per particle is shown as a function of the number of nodes: The performance and time to solution demonstrate essentially perfect scaling. Strong scaling results (blue) are for a fixed-size problem – 1024^3 particles in a 1.42Gpc box. The final number of nodes is 8192, approximately half of Titan. Optimal scaling is shown in black.***

Table 2: Weak scaling performance on Titan with $\simeq 3.2 \cdot 10^7$ particles per node (double for the last run), N_p=Number of particles, L=Boxlength. We omit two smaller runs shown in Fig. 3.

Nodes N	N_p	L [Mpc]	N*Time [sec]/Substep/N_p
512	2560^3	2880	$2.05 \cdot 10^{-6}$
1024	3200^3	3628	$1.77 \cdot 10^{-6}$
2048	4096^3	4571	$2.30 \cdot 10^{-6}$
4096	5120^3	5714	$2.37 \cdot 10^{-6}$
8192	6656^3	6857	$2.22 \cdot 10^{-6}$
16384	8192^3	9142	$2.69 \cdot 10^{-6}$
16384	10240^3	11429	$2.50 \cdot 10^{-6}$

node on the BG/Q and one rank per node on Titan), the scaling remains close to ideal, similar on both machines. In the second set of scaling tests, the grid size per rank is held constant, at approximately 200^3 for the BG/Q and 300^3 for Titan (the last Titan run was increased to 400^3 particles per rank). FFT scaling is demonstrated on the BG/Q up to 16 racks and to a size of 10240^3. The performance is remarkably stable, predicting excellent FFT performance on the largest systems and, as shown in the next section, holding up to 96 racks. For our final full runs we measured the overall timing only – the excellent scaling of the full code is proof of the FFT scaling up to 15360^3 grid sizes on more than 1.5 million cores. (Note that in these runs, we kept $N_g = N_p$, i.e., number of grid points and particles are equal. In general, $N_g \neq N_p$ is allowed.)

4.3.2 *Scaling of the Full Code up to 16384 Nodes on Titan*

We present performance data for two cases: 1) weak scaling on Titan with up to 16,384 nodes; and 2) strong scaling on up to 8,192 nodes with a fixed-size simulation problem. Timing results are obtained by averaging over 15 substeps.

Weak Scaling: We ran with 32 million particles per node in a fixed (nodal) physical volume of $(360\ \mathrm{Mpc})^3$, representative of the particle loading in actual large-scale simulations (the GPU version of the code was run with one MPI rank per node). The results are shown in Fig. 3 and Table 2. The last row shows the timing for a 1.1 trillion particle run, where we have kept the volume per node the same but increased the number of particles per node by a factor of two to 64.5 million. This is by far the largest cosmological simulation benchmark carried out on a GPU-accelerated system, demonstrating essentially perfect weak scaling with respect to time to solution.

Strong Scaling: Many-core based architectures are tending inexorably towards a large number of heterogeneous cores per node with a concomitant decrease in the byte/flop ratio, a defining characteristic of exascale systems. For these future-looking reasons – anticipating the strong-scaling barrier for large-scale codes – and for optimizing wall-clock at fixed problem size, it is important to establish the robustness of the strong scaling properties of the HACC algorithms.

We ran a fixed-size 1024^3 particle problem while increasing the the number of nodes from 32 to 8192, almost half of Titan. The results are shown in Fig. 3. For the run with the smallest number of nodes, we utilize ∼30% of the available

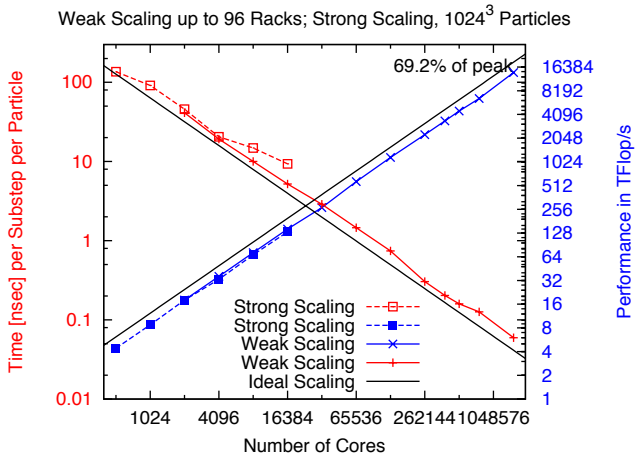

Figure 4: ***Weak and strong scaling on the BG/Q; time per substep per particle (red) and overall performance (blue), as a function of the number of cores. The offset black lines indicate ideal scaling. Weak scaling (solid lines with crosses) is reported for ∼2 million particles per core for up to 96 racks. Performance and time to solution demonstrate essentially perfect scaling. Strong scaling (dashed lines, boxes) follows the same set up as for Titan (Figure 3), the number of cores going from 512 to 16384. The timing scales nearly perfectly to 8192 cores, then degrades slightly; performance stays high throughout.***

CPU memory (32 GB per node), for the largest, we use less than 1%. Up to 512 nodes, HACC strong-scales almost perfectly, after which the scaling degrades somewhat. This is not surprising – at this low particle loading the GPUs lack the computational work to hide the particle transfer penalty to the CPU. A utilization of less than 1% of the available memory is not a real-world scenario, yet even at this value, HACC performs extremely well.

4.3.3 *Scaling of the Full Code up to 96 Racks of the BG/Q*

We present data for two cases: 1) weak scaling at 90% parallel efficiency with up to 1,572,864 cores (96 racks); 2) strong scaling with up to 16,384 cores with a fixed-size problem to explore future systems with lower memory per core. Timing results are obtained by averaging over 50 substeps.

Weak Scaling: We ran with 2 million particles per core, a typical particle loading in actual large-scale simulations on BG/Q systems. Tests with 4 million particles per core produce very similar results. As demonstrated in Fig. 4 and Table 3, weak scaling is ideal up to 1,572,864 cores (96 racks), where HACC attains a peak performance of 13.94 PFlops and a time per particle per substep of ~ 0.06 ns for the full high-resolution code. This problem, with 3.6 trillion particles, is more than a factor of 10 larger than the Millennium XXL simulation [3]. The time to solution is set by the science use requirement, i.e., running massive high-precision HACC simulations on a production basis – within days rather than weeks. The performance achieved allows runs of 100 billion to trillions of particles in a day to a week of wall-clock time.

Strong Scaling: The problem set-up follows that on Titan, using 512 to 16384 cores, and spanning a per node memory utilization of $\sim 57\%$, a typical production run value, to as low as 7%. The actual utilization scales by a factor of

Table 3: Performance Summary of Weak Scaling Tests on the BG/Q ($\simeq$ 2,000,000 Particles per Core), N_p=Number of particles, L=Boxlength. We omit two smaller runs shown in Fig. 4.

Cores C	N_p	L [Mpc]	Peak [%]	C*Time /Substep/N_p
8,192	2560^3	2880	68.75	$8.21 \cdot 10^{-5}$
16,384	3200^3	3628	68.50	$8.50 \cdot 10^{-5}$
32,768	4096^3	4571	69.02	$9.44 \cdot 10^{-5}$
65,536	5120^3	5714	68.64	$9.59 \cdot 10^{-5}$
131,072	6656^3	6857	69.37	$9.70 \cdot 10^{-5}$
262,144	8192^3	9142	67.70	$7.96 \cdot 10^{-5}$
393,216	9216^3	9857	67.27	$7.99 \cdot 10^{-5}$
524,288	10240^3	11429	67.46	$8.36 \cdot 10^{-5}$
1,572,864	15360^3	16614	69.22	$9.93 \cdot 10^{-5}$

8, instead of 32, because on 16384 nodes we are running a very small simulation volume per rank with high overloading memory and compute cost. Given that the algorithms are designed to run at $> 50\%$ of per node memory utilization to a factor of 4 less ($\sim 15\%$), the strong scaling, as depicted in Fig. 7 is impressive. It stays near-ideal throughout, slowing down at 16384 cores, only because of extra computations in the overloaded regions. This test demonstrates that the HACC algorithms will work well in situations where the byte/flop ratio is significantly smaller than the optimal plateau for the BG/Q.

5. THE FUTURE

HACC framework development is tightly coupled to future architectures; in particular the two-level design naturally maps the lower level (short-range solvers) to the individual node architecture. Because of this feature, architecture co-design with HACC kernels is particularly straightforward. Conversely, knowledge of the nodal architecture allows for ease of optimization and algorithmic choices within the framework.

The choice of programming models is also connected to this two-level structure. It appears very likely that in the near future ('pre-exascale' systems), the number of nodes – distinct pieces of hardware directly connected to the main system network – will not be too different from 100,000. HACC has already demonstrated running at more than 1.5 million MPI ranks, and it is unlikely that a significant improvement in this number will be needed. Programming models at the node level will evolve, and, as already demonstrated, HACC can easily adapt to these.

Future nodes will express unprecedented levels of concurrency; possibly thousands of independent threads. The grand challenge for pre-exascale applications will be how to adapt to this change. In principle, HACC has the potential to utilize a large number of independent streams effectively; precisely how to do this is a key focus area. In terms of the bytes/flop ratio, given a pre-exascale system with notional numbers of 100 PFlops and 10 PB of total memory, the results presented here show that HACC is ready for such a machine today.

There are a number of technical issues such as complex memory hierarchies (e.g., NVRAM) and bandwidth, power management, and resilience technologies that we continue to monitor. HACC has been listed as a benchmark for gathering information regarding future supercomputers at Argonne, Livermore, and Oak Ridge. This provides an opportunity to stay abreast of the latest advances; because HACC does not rely on 'black box' libraries or packages, it retains the key advantage of allowing optimization to be a continuous process.

Many of the ideas and methods presented here are relatively general and can be re-purposed to benefit other HPC applications, especially in areas such as accelerator beam dynamics and plasma simulations, particle transport codes, and for molecular dynamics simulations.

An important aspect of HACC not mentioned so far is an associated parallel in situ analysis and visualization framework called CosmoTools that runs in tandem with HACC simulations to perform 'on the fly' analysis (computation of summary statistics, halo finding, halo merger trees, etc.) and data reduction tasks. The very large simulations undertaken by the HACC framework make having such a capability an absolute requirement, as extensive post-processing of the raw data outputs is almost impossible. Future development of HACC will also involve associated development of CosmoTools.

Science results from a number of HACC simulations include a suite of 64 billion particle runs for baryon acoustic oscillations predictions for BOSS (Baryon Oscillation Spectroscopic Survey) carried out on Roadrunner [36] and a high-statistics study of galaxy cluster halo profiles [4]. Several other large simulations are currently being analyzed. Next-generation HACC applications include a currently running 1.07 trillion particle simulation (Fig. 1), a suite of thirty 64 billion particle simulations for the BOSS survey spanning multiple cosmologies (5 completed), and designed to construct full-sky synthetic catalogs out to a redshift depth of $z \sim 0.8$. These simulations will form a template for simulation efforts for future surveys. Perhaps the most significant future science target for HACC are predictions for the Large Synoptic Survey Telescope (LSST) project [1], which will require high mass resolution simulations in very large cosmological volumes.

Acknowledgment

We are indebted to Bob Walkup for running HACC on a prototype BG/Q system at IBM and to Dewey Dasher for help in arranging access. At ANL, we thank Susan Coghlan, Paul Messina, Mike Papka, Rick Stevens, and Tim Williams for obtaining allocations on different Blue Gene systems. At LLNL, we are grateful to Brian Carnes, Kim Cupps, David Fox, and Michel McCoy for providing access to Sequoia. At ORNL, we thank Bronson Messer and Jack Wells for assistance with Titan. This research used resources of the ALCF, which is supported by DOE/SC under contract DE-AC02-06CH11357 and resources of the OLCF, which is supported by DOE/SC under contract DE-AC05-00OR22725.

6. ADDITIONAL AUTHORS

Additional authors: Kalyan Kumaran, Venkat Vishwanath, Tom Peterka, Joe Insley; Affiliation: Argonne National Laboratory, emails: {kumaran, venkat, tpeterka, insley}@anl.gov; David Daniel, Patricia Fasel; Affiliation: Los Alamos National Laboratory, emails: {ddd, pfj}@lanl.gov; Zarija Lukić, Affiliation: Lawrence Berkeley National Laboratory, email: zarija@lbl.gov

7. REFERENCES

[1] Abell, P.A. et al. [LSST Science Collaborations and LSST Project Collaboration], arXiv:0912.0201 [astro-ph.IM].

[2] Alimi, J.-M., V. Bouillet, Y. Rasera, V. Reverdy, P.-S. Corasaniti, I. Balmes, S. Requena, X. Delaruelle, and J.-N. Richet, arXiv:1206.2838v1 [astro-ph.CO], *Supercomputing 2012*.

[3] Angulo, R.E., V. Springel, S.D.M. White, A. Jenkins, C.M. Baugh, and C.S. Frenk, arXiv:1203.3216v1 [astro-ph.CO].

[4] Bhattacharya, S., S. Habib, K. Heitmann, and A. Vikhlinin, Astrophys. J., **766**, 32 (2013).

[5] Bryan, G.L. and M.L. Norman, 12th Kingston Meeting on Theoretical Astrophysics, proceedings of meeting held in Halifax; Nova Scotia; (ASP Conference Series # 123), D.A. Clarke. and M. Fall (1996); B.W. O'Shea, K. Nagamine, V. Springel, L. Hernquist, and M.L. Norman, Astrophys. J. Supp., **160**, 1 (2005).

[6] Chen, D., N.A. Eisley, P. Heidelberger, R.M. Senger, Y. Sugawara, S. Kumar, V. Salapura, D.L. Satterfield, B. Steinmacher-Burowin, and J.J. Parker, in SC11, Proceedings of the 2011 Int. Conf. High Performance Computing Networking Storage and Analysis (2011).

[7] Couchman, H.M.P., P.A. Thomas, and F.R. Pearce, Astrophys. J., **452**, 797 (1995).

[8] For a recent review of cosmological simulation methods, see Dolag, K., S. Borgani, S. Schindler, A. Diaferio, and A.M. Bykov, Space Sci. Rev. **134**, 229 (2008).

[9] Dubinksi, J., J. Kim, C. Park, and R. Humble, New Astronomy **9** 111 (2004).

[10] Fryxell, B., et al., Astrophys. J. Supp., **131**, 273 (2000).

[11] Gafton, E. and S. Rosswog, Mon. Not. R. Astron. Soc., **418**, 770 (2011).

[12] Habib, S., unpublished.

[13] Habib, S., V. Morozov, H. Finkel, A. Pope, K. Heitmann, K. Kumaran, T. Peterka, J. Insley, D. Daniel, P. Fasel, N. Frontiere, Z. Lukić, arXiv:1211.4864, *Supercomputing 2012*.

[14] Habib, S., A. Pope, Z. Lukić, D. Daniel, P. Fasel, N. Desai, K. Heitmann, C.-H. Hsu, L. Ankeny, G. Mark, S. Bhattacharya, and J. Ahrens, J. Phys. Conf. Ser. **180**, 012019, (2009).

[15] Hamming, R.W., *Digital Filters* (Dover, 1998).

[16] Haring, R.A., M. Ohmacht, T.W. Fox, M.K. Gschwind, P.A. Boyle, N.H. Christ, C. Kim, D.L. Satterfield, K. Sugavanam, P.W. Coteus, P. Heidelberger, M.A. Blumrich, R.W. Wisniewski, A. Gara, and G.L. Chiu, IEEE Micro **32**, 48 (2012).

[17] Hearin, A., A. Zentner, and Z. Ma, arXiv:1111.0052v2 [astro-ph.CO].

[18] Heitmann, K., P.M. Ricker, M.S. Warren and S. Habib, Astrophys. J. Supp. **160**, 28 (2005).

[19] Heitmann, K., Z. Lukić, P. Fasel, S. Habib, M.S. Warren, M. White, J. Ahrens, L. Ankeny, R. Armstrong, B. O'Shea, P.M. Ricker, V. Springel, J. Stadel, H. Trac, Comput. Sci. Dis. 1, 015003, (2008).

[20] Heitmann, K., D. Higdon, M. White, S. Habib, B.J. Williams, E. Lawrence, and C. Wagner, Astrophys. J. **705**, 156 (2009).

[21] Hockney, R.W. and J.W. Eastwood, *Computer Simulation Using Particles* (New York: Adam Hilger, 1988).

[22] IBM Blue Gene/Q Application Development Redbook; available at: www.redbooks.ibm.com/redpieces/pdfs/sg247948.pdf

[23] Jetley, P., L. Wesolowski, F. Gioachin, L.V. Kalé, and T.R. Quinn, *Supercomputing 2010*, IEEE Comp. Soc.

[24] Kim, J., C. Park, G. Rossi, S.M. Lee, and J.R. Gott III, J. Korean Astron. Soc. **44**, 217 (2011).

[25] Peebles, P.J.E., Astronomical Journal, **75**, 13 (1970).

[26] Peebles, P.J.E., *The Large-Scale Structure of the Universe* (Princeton University Press, 1980).

[27] Pfalzner, S., and P. Gibbon, *Many-Body Tree Methods in Physics* (Cambridge University Press, 1996); see also Barnes, J., and P. Hut, Nature **324**, 446 (1986); Warren, M.S. and J.K. Salmon, Technical Paper, Supercomputing 1993.

[28] Pope, A., S. Habib, Z. Lukic, D. Daniel, P. Fasel, N. Desai, and K. Heitmann, Computing in Science and Engineering **12**, 17, (2010).

[29] Power, C., J.F. Navarro, A. Jenkins, C.S. Frenk, S.D.M. White, V. Springel, J. Stadel, and T. Quinn, Mon. Not. R. Astron. Soc. **338**, 14 (2003).

[30] Shandarin, S.F., and Ya.B. Zeldovich, Rev. Mod. Phys. **61**, 185 (1989). See also, Shandarin, S., S. Habib, and K. Heitmann, Phys. Rev. D **85**, 083005 (2012).

[31] Schive, H.Y., Y.-C. Tsai, and T. Chiueh, Astrophys. J. Supp. **186**, 457 (2010)

[32] Springel, V., Mon. Not. R. Astron. Soc. **364**, 1105 (2005).

[33] Teyssier, R., A&A **385**, 337 (2002).

[34] Warren, M.S and J.K. Salmon, in *Supercomputing '93* and *Supercomputing '97*, IEEE Comp. Soc.

[35] White, M. Astron. and Astrophys. **367**, 27 (2001).

[36] White, M., A. Pope, J. Carlson, K. Heitmann, S. Habib, P. Fasel, D. Daniel, and Z. Lukić, Astrophys. J. **713**, 383 (2010).

[37] Xu, G. Astrophys. J. Supp. **98**, 355 (1995).

The submitted manuscript has been created by UChicago Argonne, LLC, Operator of Argonne National Laboratory ("Argonne"). Argonne, a U.S. Department of Energy Office of Science laboratory, is operated under Contract No. DE-AC02-06CH11357. The U.S. Government retains for itself, and others acting on its behalf, a paid-up nonexclusive, irrevocable worldwide license in said article to reproduce, prepare derivative works, distribute copies to the public, and perform publicly and display publicly, by or on behalf of the Government.